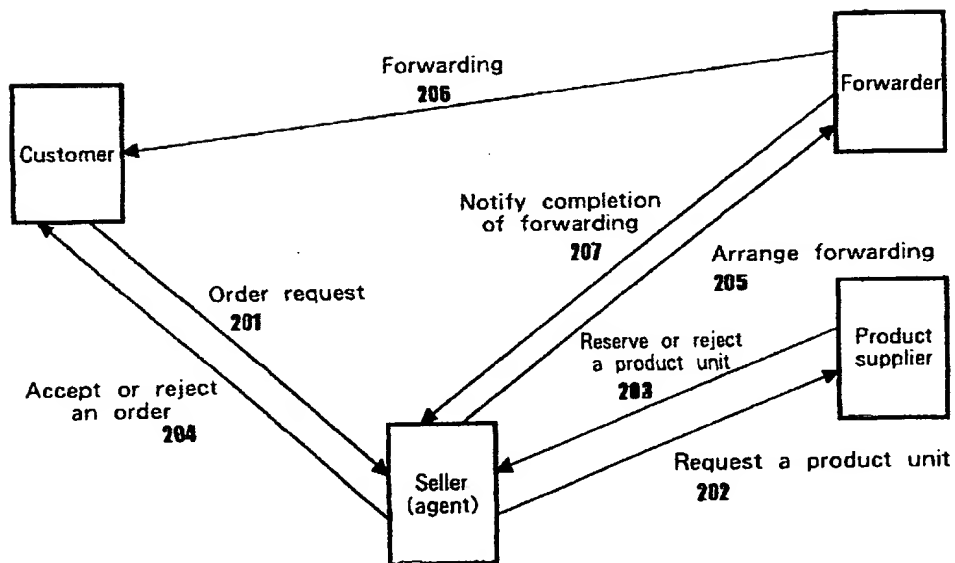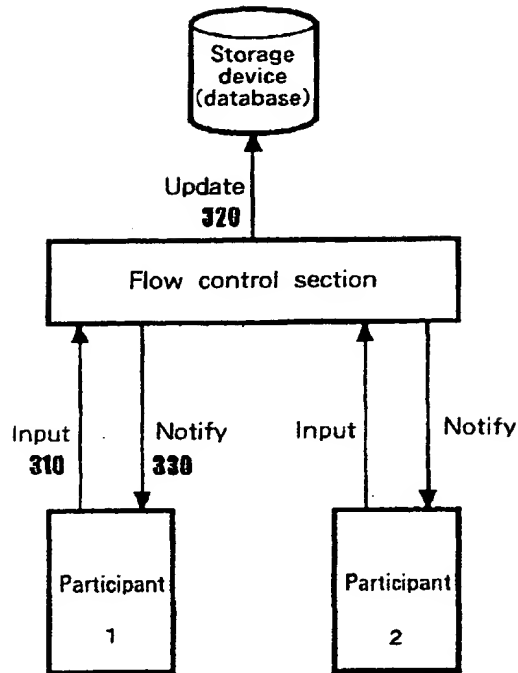Fig. 1

Typical inter − company workflow
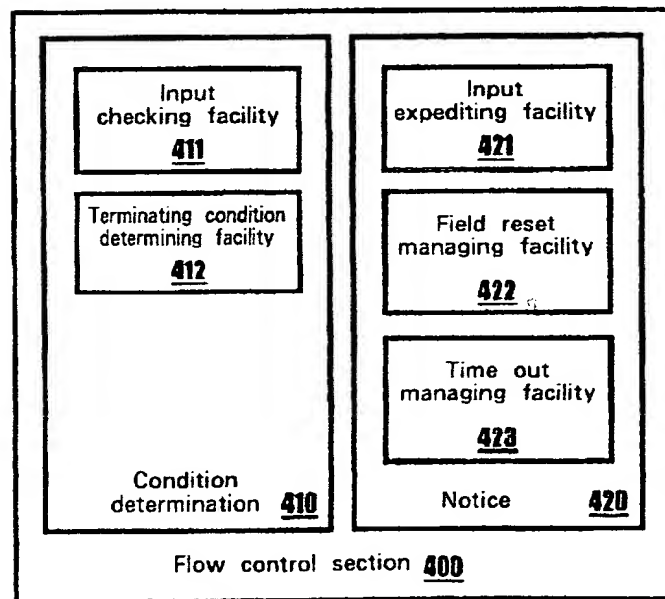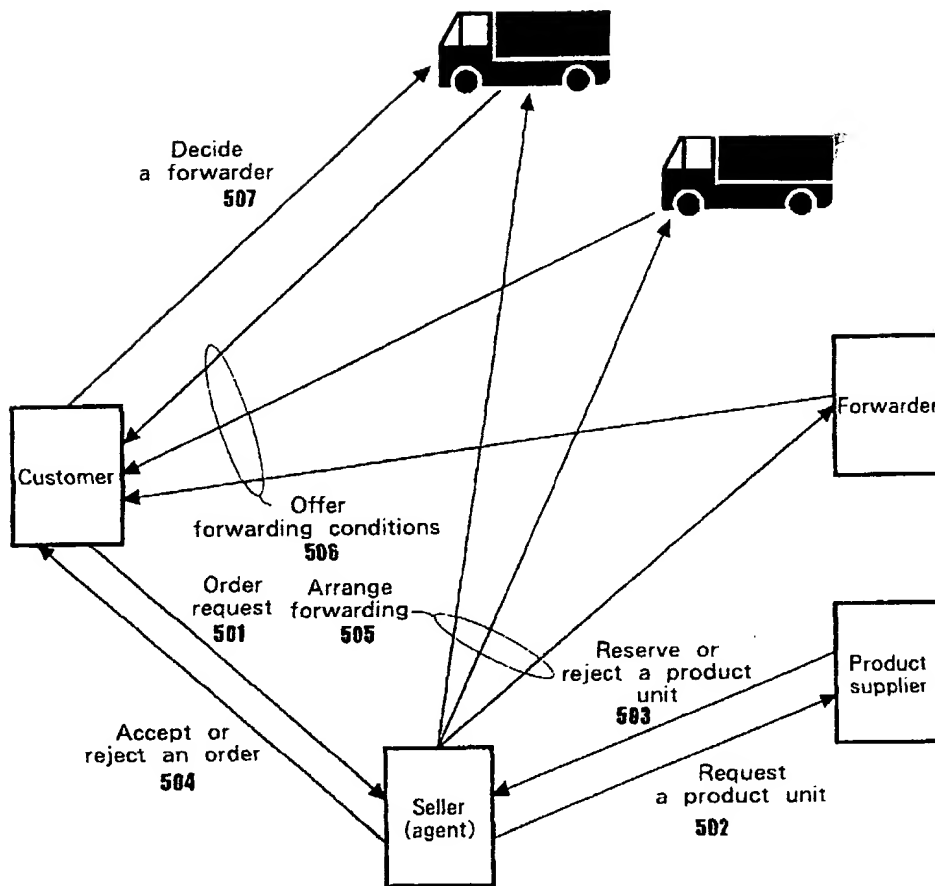
Fig. 2

Overview of a workflow controlling system

Fig. 3

Structure of a flow control section

Fig. 4

Workflow including a bid from a forwarder

Fig. 5

Contents
  Tree structure
    (Node. [value])
History
    (Time. person. action. node ID)
Access Control
    (Node ID, tag name, person, role, action, conditional expression)
Constrains
    (Conditional expression)
Dependencies
    (Depended node ID, Dependent node ID)
Termination
    (Type, conditional expression)
    Type : End or Abort

[A]   means that A is optional
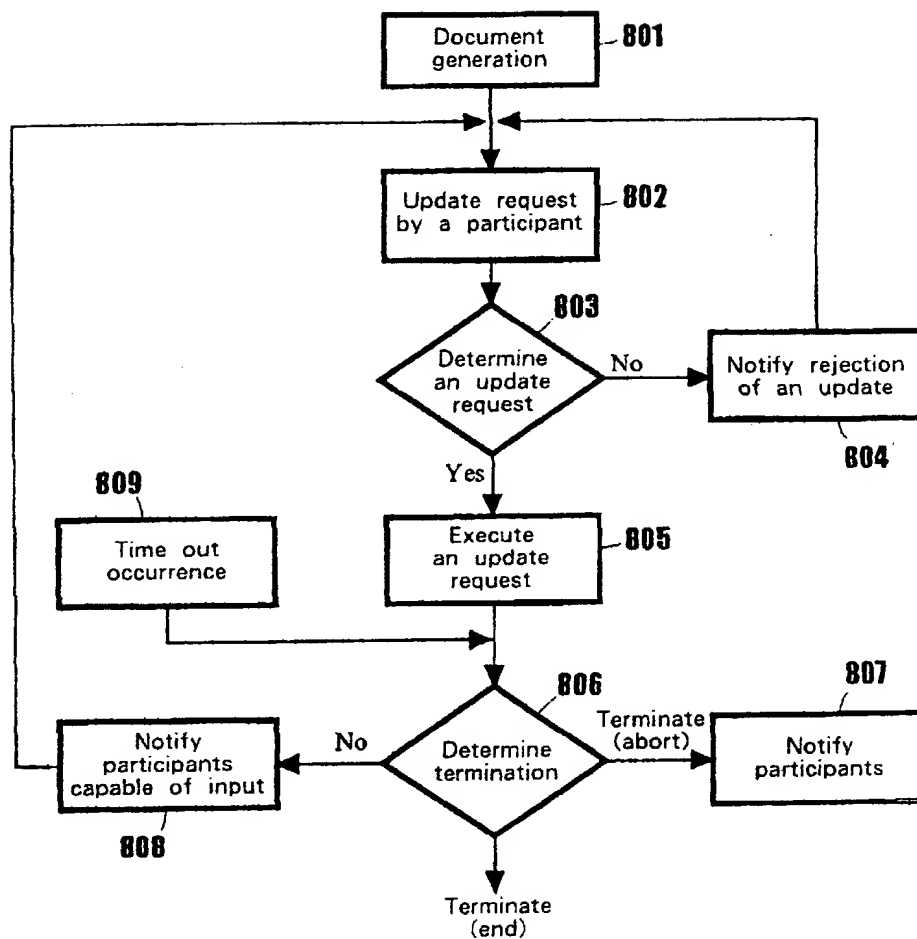
Document data structure

Fig. 6

Contents
  OrderID= "00001"
  Consumer
    ConsumerID= "ConsumerA"
    Name= "Neyama"
    Address= "Yamato-shi"
    Phone= "042-123-4567"
    DeliveryDateRequested= "21/Sep/1999"
  Product
    ProductID= "IBM Aptiva"
    Price= "99,800 yen"
    UnitID= "9.116.54.89"
  Supplier
    SupplierID= "IBM Corp."
  Transport
    Specified= "Kuroneko"
    Candidate#0= "Pelican"
      DeliveryDateOffered= "21/Sep/1999"
    Candidate#1= "Kuroneko"
      DeliveryDateOffered= "20/Sep/1999"
        ⋮

History
    14/Sep/1999:15:20:30,Runtime,w,OrderID
    14/Sep/1999:15:22:20,Neyama,w,ConsumerID
        ⋮
    14/Sep/1999:16:37:10,Pelican,cr,Candidate#0
    14/Sep/1999:16:37:20,Pelican,w,Candidate#0

Access Control
    value(ConsumerID),w,Specified
    Transport,cr,Candidate#?,(value(Specified)=nil)

Constraints
    value(DeliveryDateOffer) <= value(DeliveryDateRequested)
    timeout(isFilled(Specified)isFilled(DeliveryDateRequested),(B0)

Dependencies
    ConsumerID; OrderID
    Candidate#?; DeliveryDateRequested

Termination
    End
        value(Specified) ! nil
    Abort
        ProductID, cn
        time(Specified, w) > time(DeliveryDateRequested) +

Example of a document

Fig. 7

```
                    ┌─────────────┐
                    │  Document   │─ 801
                    │ generation  │
                    └─────────────┘
                           │
                           ▼
                    ┌─────────────┐
                    │Update request│─ 802
                    │by a participant│
                    └─────────────┘
                           │
                           ▼  803
                         ╱──────╲                    ┌──────────────┐
                        ╱Determine╲     No           │Notify rejection│
                       ╱ an update  ╲──────────────▶ │ of an update  │
                        ╲ request   ╱                └──────────────┘
                         ╲──────╱                          804
                           │
                          Yes
    809                     │
 ┌──────────┐        ┌─────────────┐
 │ Time out │        │  Execute    │─ 805
 │occurrence│        │ an update   │
 └──────────┘        │  request    │
        │            └─────────────┘
        │                   │
        └──────────────────▶│
                           ▼  806
                         ╱──────╲       Terminate    807
                        ╱Determine╲     (abort)   ┌──────────┐
 ┌──────────┐     No   ╱termination╲────────────▶│  Notify  │
 │  Notify  │◀─────────╲          ╱               │participants│
 │participants│        ╲        ╱                └──────────┘
 │capable of input│     ╲──────╱
 └──────────┘              │
    808                    ▼
                      Terminate
                       (end)
```

Operation of a flow control section

Fig. 8
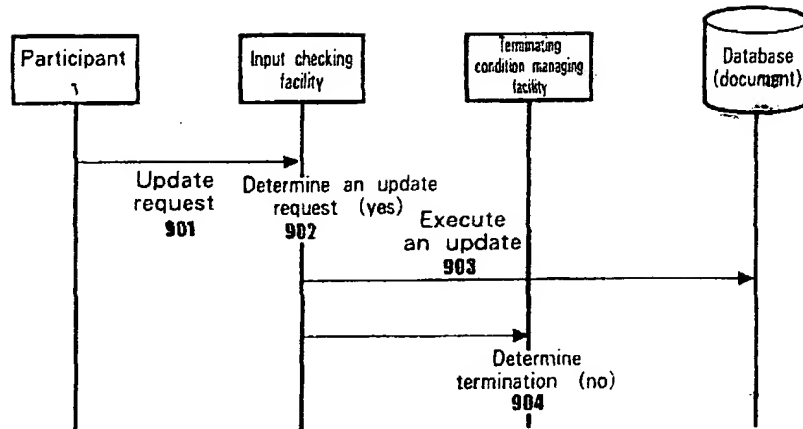
Processing flow among modules (1)

Fig. 9

Participant 1 — Input checking facility — Terminating condition managing facility — Database (document)

Update request **901** — Determine an update request (yes) **902** — Execute an update **903**

Determine termination (no) **904**

Processing flow among modules (2)

Fig. 10

Participant 2 — Participant 3 — Input checking facility — Terminating condition managing facility — Input expediting facility

1001

Notify participants capable of input **1002**

Update request **1003**

Proceed to next — Determine termination (yes) **1004**

Termination **1005**

Contents
- OrderID → 00001
- Consumer
  - ConsumerID → ConsumerA
  - Name → Neyama
  - Address → Yamato-shi
  - Phone → 046-123-4567
  - DeliveryDate Requested → 21/Sep/1999
- Product
  - ProductID → IBM-Aptiva
  - Price → 99,800yen
  - UnitID → 9.116.54.89
- Supplier
  - SupplierID → IBM Corp.
- Transporter
  - Specified → Kuruneko
  - Candidate#0 → DeliveryDateOffered → 21/Sep/1999
  - Candidate#1 → DeliveryDateOffered → 20/Sep/1999

[ ] ----- Attribute name

( ) ----- Value

Structure of contents

Fig. 11

	Node ID (Attribute name)	Parent node ID (Attribute name)	Value
T0	/	nil	nil
T1	/document	/	nil
T2	/document/contents	/document	nil
T3	/document/contents/OrderID	/document/contents	00001
T4	/document/contents/Consumer	/document/contents	nil
T5	/document/contents/Consumer /ConsumerID	/document/contents /Consumer	Neyama
T6	/document/contents/Consumer /ConsumerID/Name	/document/contents /Consumer	Ryoh Neyama
T7	/document/contents/Consumer /ConsumerID/Address	/document/contents /Consumer	Yamato-shi
T8	/document/contents/Consumer /ConsumerID/Phone	/document/contents /Consumer	046-123-4567

Representation of a tree structure of contents as a table

Fig. 12

Order	Time (sec)	Writer ID	Action	Node ID
0	0	Runtime	Write	/document/contents/OrderID
1	100	Neyama	Write	/document/contents/Consumer /ConsumerID
2	100	Neyama	Write	/document/contents/Consumer /Name
3	100	Neyama	Write	/document/contents/Consumer /Address
4	100	Neyama	Write	/document/contents/Consumer /Phone

(Action types : Create, Write, Read, Cancel)

Example of History representation

Fig. 13

```
Outline part format
    allow( <node>,<user>,<operation>)

Example of rules
  Rule 1
      allow( ?Node, ?User, "+w") <--
          isPath( ?Node, "/document") and
          hasRole( ?User, "Consumer").

  Rule 2
      allow( ?Node, ?User, "+w") <--
          isPath( ?Node, "/ProductID") and
          hasRole( ?User, "Consumer") and
          isCreator( ?User, ?Node).
```

Example of Access Control representation

Fig. 14

Constraints 1

Contents : member(TransportSpecified, CompanyID)

Internal representation :
    getValue('TransportsSpecified',V1) and
    getValueList('CompanyID',V2) and
    member(V1, V2)

Constraints 2

Contents : DeliveryDateOffered <= DeliveryDateRequested

Internal representation :
    getValue('DeliveryDateRequested',V1) and
    getValue('DeliveryDateOffered',V2) and
    V1 <= V2

Example of Constraints representation

Fig. 15

Depended node ID	Dependent node ID
ProductID	UnitID
UnitID	TransportInfo
TransportInfo	TransportSpecified

Example of Dependencies representation

Fig. 16

Example of an end

     (1) isFilled('TransportSpecified').

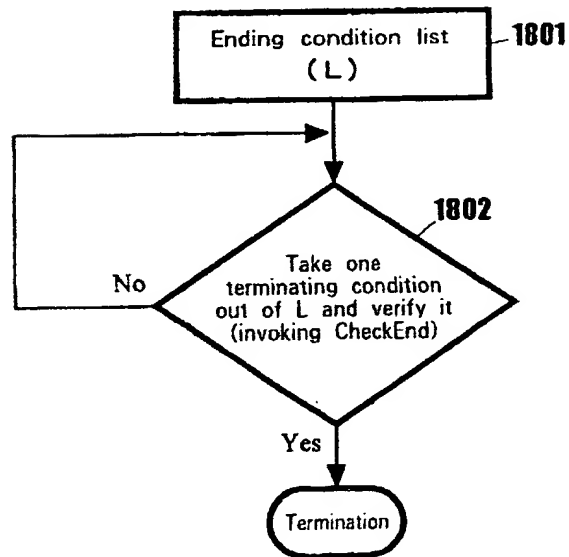Examples of an abort

     (2) isCancelled('ProductID').
     (3) timeout(
         isSpecified('ProductID'),
         isSpecified('TransportSpecified'),
         180).

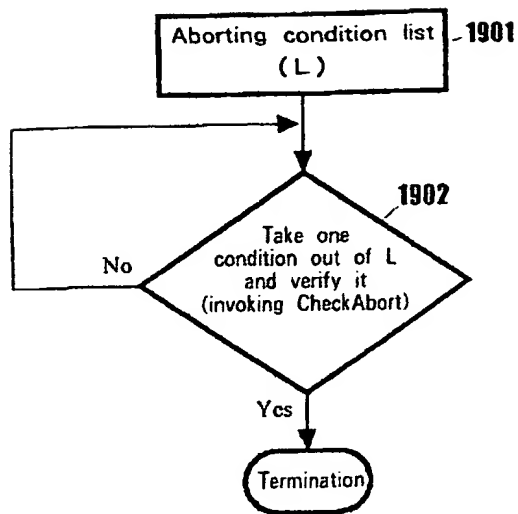**Example of Termination representation**

Fig. 17

Ending condition list
(L) — **1801**

1802

Take one
terminating condition
out of L and verify it
(invoking CheckEnd)

No

Yes

Termination

End determination

Fig. 18

Aborting condition list —1901
(L)

1902
Take one
condition out of L
and verify it
(invoking CheckAbort)

No

Yes

Termination

Abort determination

Fig. 19

Input condition C —2001

Retrieve one data set DS
meeting condition C —2002

2003
Retrieval
successful ?

No

Output a list —2004

Yes

Add DS to the list —2005
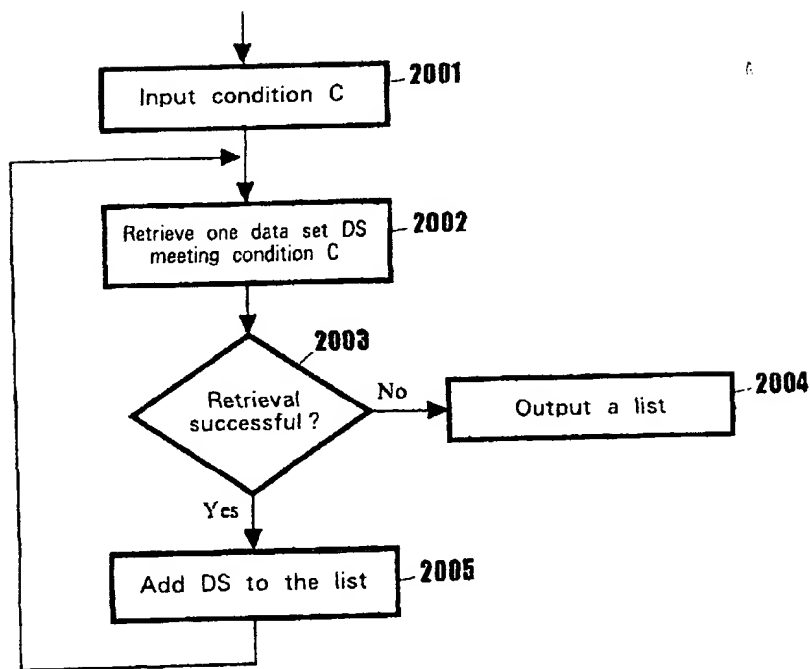
Processing for finding all elements

Fig. 20

Execution of
an update request

Execute an update —2101

Cancellation
request ? —2102

Yes → Reset dependent
fields and notify —2103

No

Register/delete
time out —2104

Determine
termination
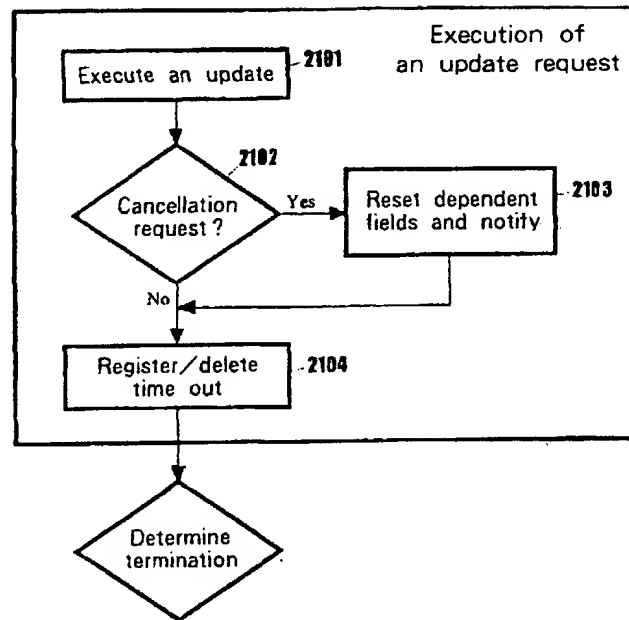
Details of execution of an update request

Fig. 21

Participant
2

Participant
1

Input checking
facility

Field reset
managing facility

Database
(document)

Update
request
2201

Determine an update
request (yes)
2202

Execute
an update
2203

Cancellation
request ? (yes)
2204

Notify
2206

Reset dependent
fields
2205
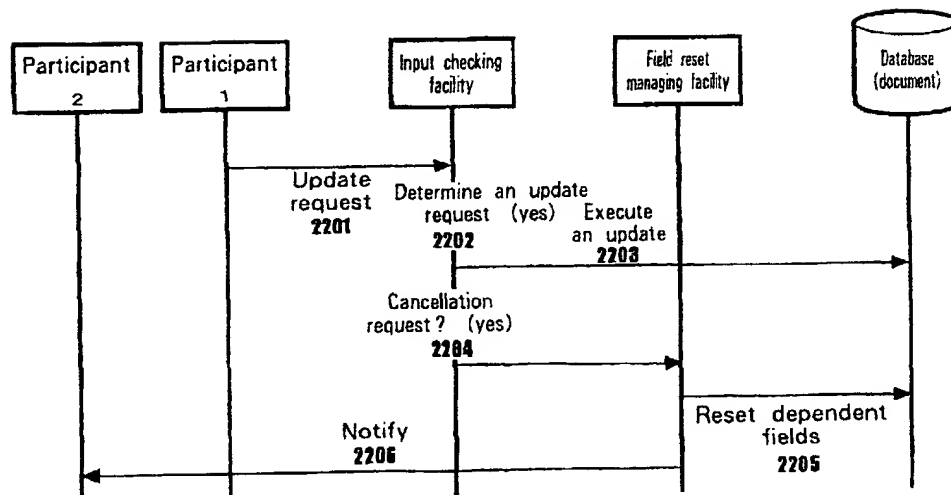
Details of update processing

Fig. 22

Perform the following initialization
based on inputted node (N)

CurrentList = {N}
AnswerList = []

— 2301

Seek a set of nodes dependent
on a node in CurrentList

(TmpList)

— 2302

Perform the following assignment

AnswerList = CurrentList
CurrentList = TmpList

— 2303

2304

Is CurrentList
empty ?

No

Yes

AnswerList

— 2305
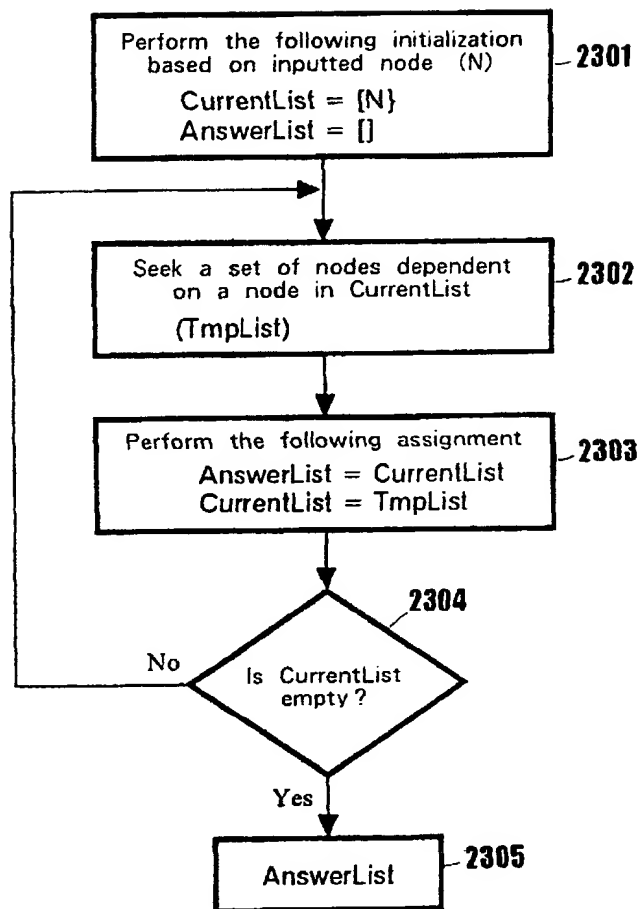
Processing for finding dependent nodes

Fig. 23

Registration and
deletion of time out

Extract any clause
including time out —2401

Extract any time − out
predicate in the clause —2402

2403

Is the first
argument
true ?

No

Yes

Register on a time out
management table —2404

Deleted from a time out
management table —2405
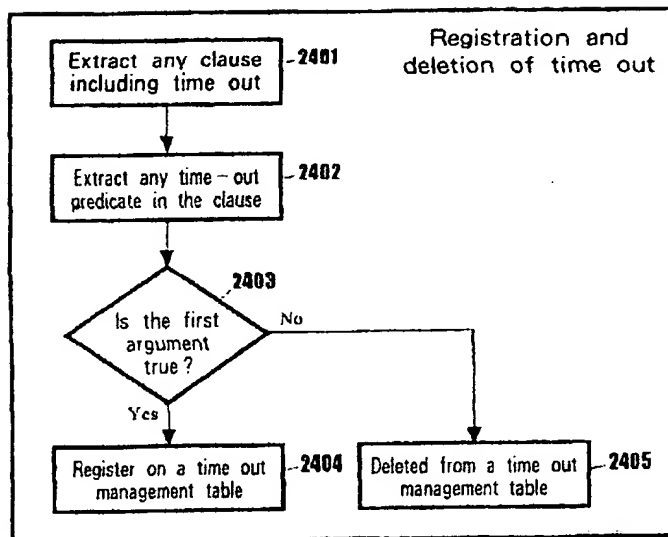
Details of registration and deletion of time out

Fig. 24

| Participant 2 | Participant 1 | Time out managing facility | Terminating condition managing facility | Input expediting facility |

Register／delete
time out
2501

Time out
occurrence
2502

Determine
termination (no)
2503

Notify participants capable of input | 2504
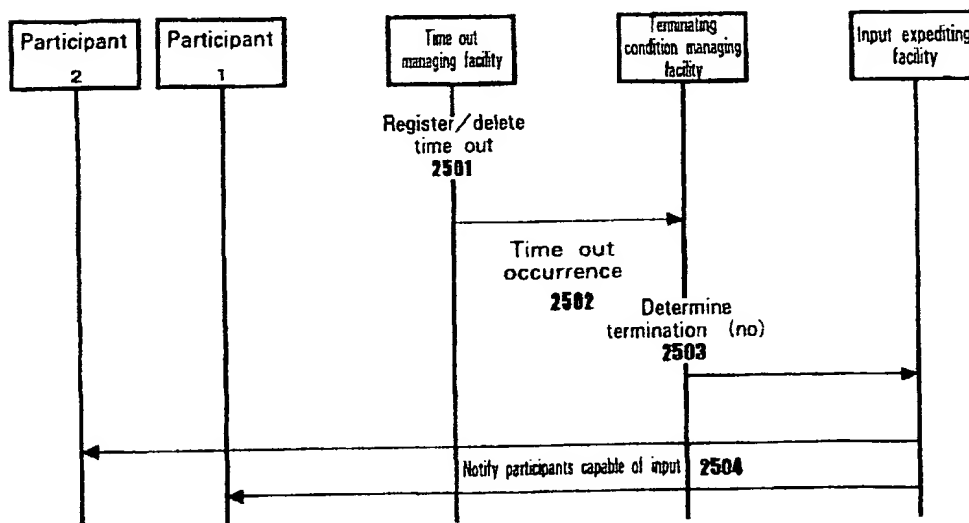
Processing after time out registration and occurrence

Fig. 25